

IWM :

Some Undocumented Features

Mike A.

April 10, 1984

During the course of debugging the hardware and firmware for the LIRON controller, we have come across two aspects of the operation of the IWM about which were unaware.

Indexed Store Problem

This quirk involves writing data in fast, asynchronous mode with the 6502. The problem arises when writing data to the IWM.

During asynchronous writes, the shift register is loaded from the write latch every byte time. For 9 FCLK periods after the shift register load, writes to the write latch are disabled. Data can be written and re-written to the latch after that time and up until the next shift register load.

On the Apple // & //e it is customary to access the IWM in an expansion slot with indexed loads and stores. The indexed store causes two accesses to the IWM. The first, the so-called "false read" is a fetch from the effective address, and the second access is the store which has valid data. The false read causes the IWM write latch to be loaded with floating bus data. The actual write cycle attempts to load the valid data into the write latch, but is unsuccessful since a reload of the latch is not allowed until 9 'FCLKS' after the false read.

The result is if you just start writing bytes to the IWM every 16 microseconds, you will transmit nothing but garbage data.

Solution:

Since the IWM only disallows reloading the latch around the time that the latch data is loaded to the shift register, it is possible to get around this problem by being careful about when you load data into the latch. If the false read occurs outside the 9 FCLK window after the shift register load, then the valid store access will replace the erroneous data in the latch with the valid data. Therefore, writing out a dummy byte about a half byte time before sending the real data solves the problem. This is the solution we are currently using on LIRON.

Another solution involves ensuring that the indexed store involves a page boundary crossing, in which case the false read does not involve the effective address (IWM location), but an address in the previous page in memory.

IWM Register Access

When reading a register on the IWM, the load must be from a bit clearing location, such as L6CLR, for valid data to be loaded. Stores for writing to registers must always be done to bit setting locations, such as L6SET.

Example:

```
20E0:           187 *
20E0:A0 0E      (2) 188     ldy #14          ;Fixed # of bytes
20E2:B0 80 C0    (4) 189 rdh3   lda 16clr,x ;If byte ready, grab it
20E5:10 FB 20E2(3) 190     bpl rdh3
20E7:99 10 03    (5) 191     sta sendbuf,y
20EA:88          (2) 192     dey
20EB:10 F5 20E2(3) 193     bpl rdh3
20ED:           194 *
20ED:           195 * Wait for /BSY to go low
20ED:           196 *
20E0:B0 E0 C0    (4) 197     lda 16set,x
20F0:B0 EE C0    (4) 198 rdt4   lda 17clr,x
20F3:30 FB 20F0(3) 199     brai rdh4
```

In this example, a number of bytes are read from the IWM (assume that L7 and L6 are cleared previously), and then the sense line is polled until it is low. It is not possible to change line 198 (underlined) to a LDA l6set,x, even though the setting of L6 and L7 are undisturbed. A read from the status register must always be from L7CLR or some other clear location.

This behavior, which is rumored to also appear in the operation of the disk J[
], poses no problems; it is simply something the programmer should be aware of.